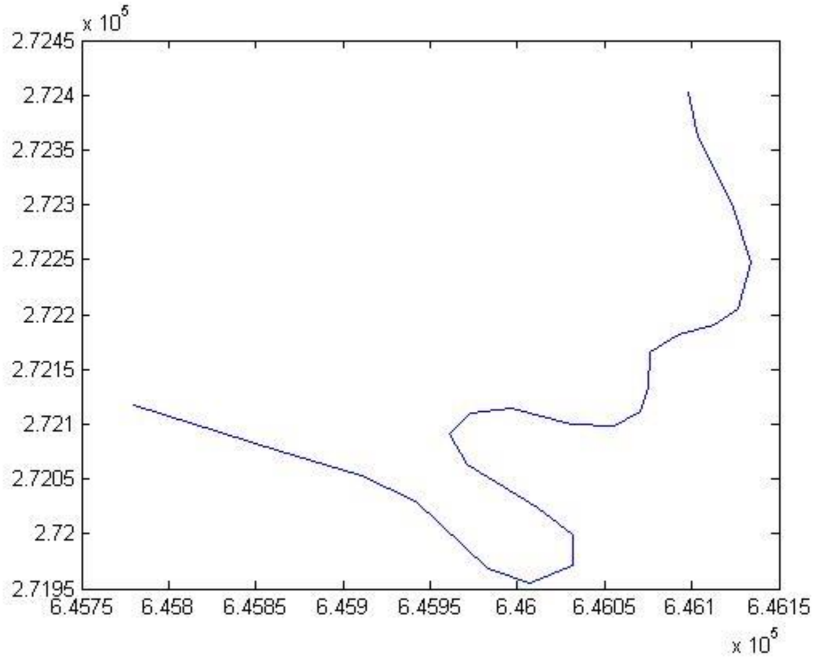
TÖBBÉRTÉKŰ GÖRBÉK INTERPOLÁCIÓJA

A spline interpoláció jól alkalmazható olyan görbék közelítésére is, amelyeknél a függvény egy adott x pontban több y értéket is felvehet (pl. körvonal esetén). Ilyen esetben a görbe paraméteres alakját használjuk. Paraméter lehet például a pont sorszáma, vagy a görbe ívhossza is. Nézzünk most ez utóbbira egy példát!

Felmérték a Visegrádi várba vezető szerpentinút közel egy kilométeres szakaszán a tengelyvonalat. Ábrázoljuk a felmérés eredményeit, majd illesszünk spline görbét a pontokra és határozzuk meg a kezdőponttól számított 500 m-es útszelvény EOV koordinátáit! Az interpolációhoz válasszuk paraméternek a görbe pontjainak a távolságát (ívhosszát). Az adatokat töltsük be a ***visegrad.txt*** fájlból. Ebben EOV vetületben vannak a felmért pontok Y,X koordinátái.

```
> % Többértékű görbék interpolációja
> clc; close all; clear all;
> adat = load('visegrad.txt')
> x = adat(:,1); y = adat(:,2);
> figure(1)
> plot(x,y)
```



Az ábrán láthatjuk, hogy vannak olyan helyek, ahol egy adott x koordinátához több y érték is tartozik, hagyományos függvény interpolációval nem tudnánk rá görbét illeszteni. Paraméteres alakban viszont megoldható a feladat. Olyan paramétert kell választanunk, ami minden pont esetében egyértelmű megoldást ad. Legyen most ez a paraméter a görbe ívhossza, amit a felmért pontok közötti távolsággal fogunk közelíteni. Ezután két függvényillesztést hajtunk végre, egyet az x koordinátára az ívhossz függvényében, egyet pedig hasonlóképp az y koordinátára.

Számítsuk ki először a pontok x,y irányú koordináta különbségeit ($\Delta x, \Delta y$)! Ezt egyszerűen megtehetjük a numerikus **diff** parancsot használva. Ez egy vektor esetében a szomszédos elemek különbségeit adja vissza (értelemszerűen *n* elem esetén *n-1* értéket). Ebből már számíthatunk távolságokat Pitagorasz-tétellel (a kezdőpontnak saját magától mért távolsága 0), majd ezeket folyamatosan összegezve a **cumsum** paranccsal megkapjuk a kezdőponttól mért távolságokat.

```
> %  Paraméter: a görbe ívhossza, távolságokkal közelítve
> % x,y irányú koordináta különbségek a pontok között:
> dx = diff(x)
> dy = diff(y)
> % távolságok:
> t0 = [0; sqrt(dx.^2+dy.^2)]
> % folyamatosan összegezve:
> t = cumsum(t0)
```

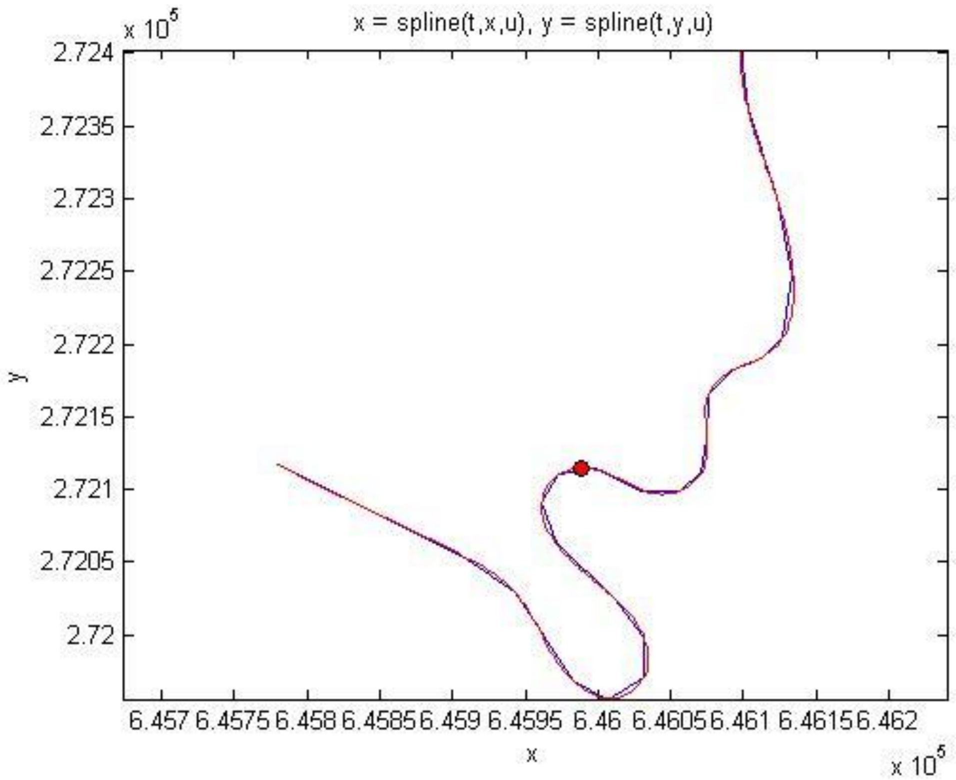
Illesszünk spline görbét az x és y értékekre az ívhossz függvényében!

```
> %  Az interpolációs függvények x-re és y-ra
> xt=@(u) spline(t,x,u);
> yt=@(u) spline(t,y,u);
> %  Ábrázoljuk az eredményt
> hold on;
> fplot(xt,yt,[0,t(end)],'r');
```

Mi lesz az EOV koordinátája az 500-as szelvény tengelypontjának?

```
> % Milyen koordináta tartozik az 500 m-es szelvényhez?
> format long
> x500 = xt(500) % 6.459890032622117e+05
> y500 = yt(500) % 2.721148370593938e+05
> format short
> plot(x500,y500,'ko','MarkerFaceColor','r')
```

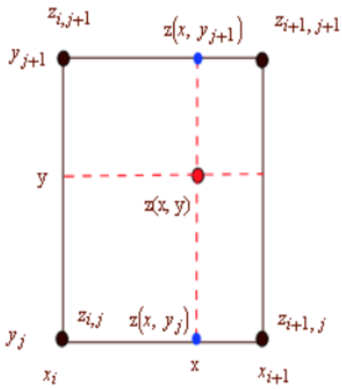
Tehát a felmérés kezdőpontjától az 500-as szelvény EOV Y koordinátája 645 989.00 m, az EOV X koordinátája pedig 272 114.84 m.



KÉTVÁLTOZÓS INTERPOLÁCIÓ SZABÁLYOS RÁCSON ADOTT PONTOK ESETÉN[1]

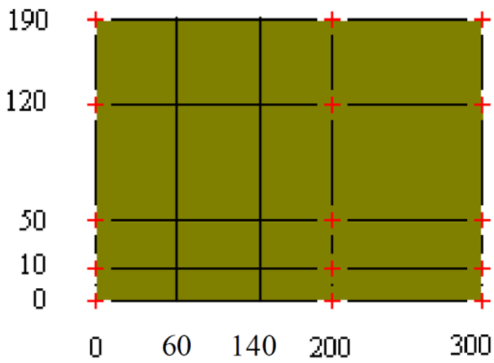
Kétváltozós esetben a legegyszerűbb interpolációs módszer a szabályos rácshálóban adott pontoknál alkalmazható bilineáris interpoláció. Az interpoláció három egyváltozós interpolációs lépésből állhat:

1) Interpoláció az $y = y_j$mentén vízszintesen: $z(x, y_j)$,
2) interpoláció az $y = y_{j+1}$ mentén vízszintesen: $z(x, y_{j+1})$,
3) végül a harmadik interpoláció az $x = állandó$mentén függőlegesen, felhasználva az előbbi két interpoláció eredményét: *z*(*x*, *y*).



Megjegyzés: A fentiek csak a bilineáris interpoláció lépései, azonban a *j-1*, *j+2* stb. pontok bevonásával magasabb fokú interpoláció is lehetséges.

Nézzünk egy tereprendezési példát a fentiekre! A terep magasságait egy szabályos rács pontjaiban mérésekkel határozták meg. A mérési pontok koordinátái és a mért magasságok az alábbiak (a magasságok a ***terep.txt*** fájlból beolvashatók):



$$Z = \begin{bmatrix} 135.0 & 131.7 & 136.4 & 139.8 & 119.8 \\ 134.8 & 133.0 & 139.7 & 135.5 & 120.0 \\ 124.4 & 130.0 & 140.0 & 139.2 & 122.3 \\ 131.1 & 133.8 & 138.1 & 137.7 & 121.1 \\ 133.2 & 137.1 & 143.0 & 135.0 & 123.3 \end{bmatrix}$$

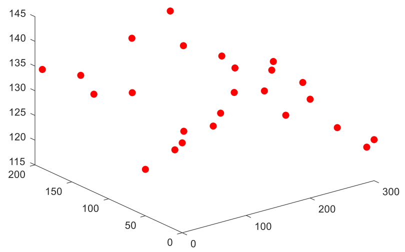
Mekkora a terep magassága a (100,100) pontban? Ábrázoljuk ebben a pontban a Ny-K irányú metszetet! Mennyi földet kell hozni vagy elvinni, ha a fenti mérési eredményekkel rendelkező domborzatot 135 m magasságú sík tereppé szeretnénk átalakítani?

A megoldáshoz elő kell állítanunk a rácspontok x,y koordinátáit is és beolvasni a hozzájuk tartozó z értékeket. Figyeljünk arra, hogy a matematikai koordináta rendszer kezdőpontja a bal alsó sarokba esik, a mátrixok kezdőpontja (első sor, első oszlop) pedig a bal felső sarokba! A koordináta rendszerek eltérése miatt itt az y koordinátákat fentről lefelé kell megadjuk!

```
> % Tereprendezes
> clc; clear all; close all;
> % A magasság értékek beolvasása
> Z = load('terep.txt')
> %  A rácspontok koordinátái (figyeljünk az adatok sorrendjére!)
```

[1] Felhasználva: Paláncz Béla (2012): Numerikus módszerek példatár + előadás fóliák

```
> x=[0 60 140 200 300];
> y=[190 120 50 10 0];
> % Rácsháló előállítása
> [X,Y]=meshgrid(x,y)
```

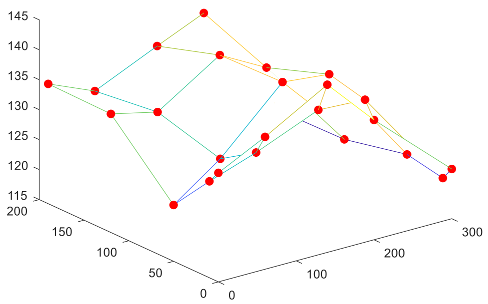


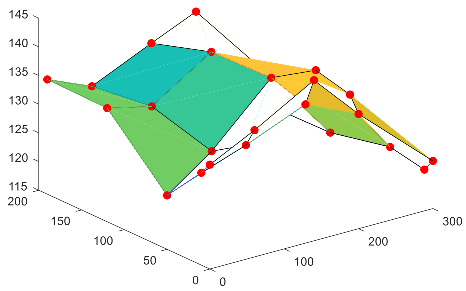
Ezzel előállt egy rácsháló x,y,z koordinátákkal. Jelenítsük meg a mért pontokat 3 dimenzióban (**plot3**)!

```
> figure(1)
> plot3(X,Y,Z,'ro','MarkerFaceColor','r')
```

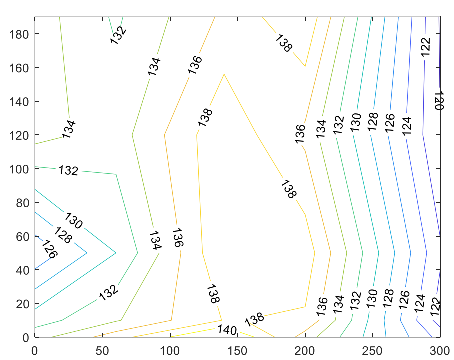
Szemléletesebben is meg lehet jeleníteni a domborzatot rácshálóként (**mesh**), vagy felületként (**surf**). Nézzük meg ezeket is!

```
> hold on; mesh(X,Y,Z)
> surf(X,Y,Z)
```





Másik gyakran használt lehetőség a szintvonalas ábrán történő megjelenítés (**contour**), ahol opcióként megadhatjuk azt is, hogy melyik szintvonalak jelenjenek meg (pl. most állítsuk be, hogy 120 és 140 méter között 2 méterenként legyenek szintvonalak), illetve bekapcsolhatjuk a szintvonalak feliratozását is a **set** paranccsal, ha két kimenettel hívjuk meg a **contour**-t, és a második kimenet tulajdonságait módosítjuk.



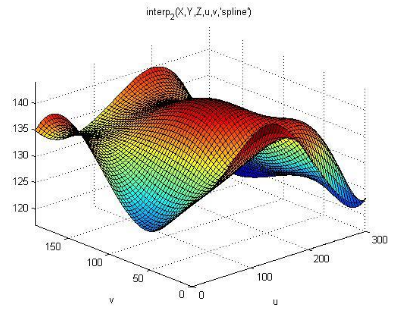

```
> figure(2)
> [C,h] = contour(X,Y,Z,120:2:140)
> set(h,'ShowText','on')
```

A fenti parancsok csak megjelenítésre szolgálnak, ebből még nem tudjuk meghatározni egy tetszőleges pontban a terep magasságát, ahhoz interpolációra (vagy regresszióra) lesz szükség.

Illesszünk a pontokra interpolációs felületet! Az interpolációhoz egy független változó esetén eddig az **interp1** parancsot használtuk, most 2 független változó esetén az **interp2** parancs használható. Az **interp2** csak rácshálóban megadott pontok/adatok esetén alkalmazható (a rácspontok egyenlő távolsága nem követelmény, csak a ’cubic’ módszer alkalmazásakor). Több különböző módszerrel is meghívható: 'nearest' - legközelebbi szomszéd, 'linear' - bilineáris interpoláció, 'spline' - spline interpoláció, 'cubic' - 2D köbös spline interpoláció (bicubic), csak egyenletes rácstávolság esetén alkalmazható, ellenkező esetben spline interpoláció kerül helyette alkalmazásra.

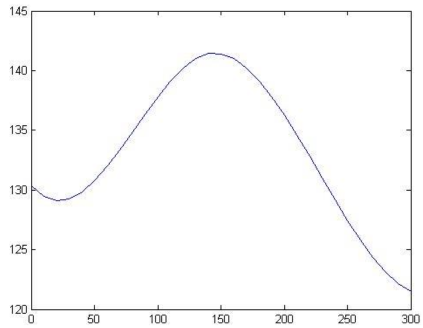
Rajzoljuk fel az interpolációs felületet az illesztett függvényt felhasználva (**fsurf** vagy **ezsurf**)!

```
> % spline interpoláció
> F=@(u,v) interp2(X,Y,Z,u,v,'spline');
> % A terep megjelenítése
> figure(3);
> fsurf(F,[0,300,0,190])
> % Ellenőrzés
> F(0,0) % 133.2000
> F(300,190) % 119.8000
> % Terep magassága a 100,100 pontban?
> F(100,100) % 137.8079
```



A (0,0) pontban a magasság 133.2 volt, a (300,190)-ben pedig 119.8, az interpoláció eredményeképpen visszakaptuk ezeket az értékeket. A (100,100) pont magassága spline interpolációval 137.8 m-re adódott. Próbaképp cseréljük le a 'spline' módszert a 'nearest'-re, 'linear'-ra, hogy lássuk, mi történik ezekben az esetekben!

Rajzoljuk fel a Ny-K irányú metszetet ebben a pontban. Ehhez vegyünk fel 50 pontot az y=100 keresztmetszetben a $P_1(0,100)$ és a $P_2(300,100)$ pontok között! Használjuk az előbb meghatározott függvényt a magasságok kiszámítására!




```
> % Ny-K irányú terepmetszet
> x0 = linspace(0,300,50)
> y0 = linspace(100,100,50)
> % vagy: y0 = ones(1,50)*100
> z0 = F(x0,y0)
> figure(4); plot(x0,z0);
```

Oldjuk meg a feladat második részét is, hogy mennyi földet kell hozni vagy elvinni, ha a fenti mérési eredményekkel rendelkező domborzatot 135 m magasságú sík tereppé szeretnénk átalakítani? Ehhez számítsuk ki a terep és a tengerszint által határolt földtömeg mennyiségét és a 135 méteres vízszintes sík és a tengerszint által határolt földtömeg mennyiségét a megadott téglalapra. A kettő különbsége fogja megadni a tereprendezéshez szükséges föld mennyiségét. A terep által határolt földtömeg mennyiségét az alábbi kettős integrállal (**integral2**) számíthatjuk:

$$V_t = \int_0^{190} \int_0^{300} F(x,y)\ dx\ dy$$

A vízszintes z=áll. sík által határolt földtömeg mennyisége egy téglatest térfogata: $V_{sz} = z \cdot 300 \cdot 190$.

```
> % Terep alatti földtérfogat téglalap tartományon kettős integrállal
> Vt=integral2(F,0,300,0,190) % 7.6140e+06
> % Vízszintes sík által határol földtérfogat
> Vs=135*300*190 % 7695000
> % A tereprendezéshez szükséges földtérfogat
> Vr=Vs-Vt % 8.0983e+04
```

Mivel a tervezett vízszintes sík által határol térfogat nagyobb volt, mint a terep alatti, ezért feltöltésre van szükség, méghozzá 80983 m^3 földre.

KÉTVÁLTOZÓS REGRESSZIÓ[2]

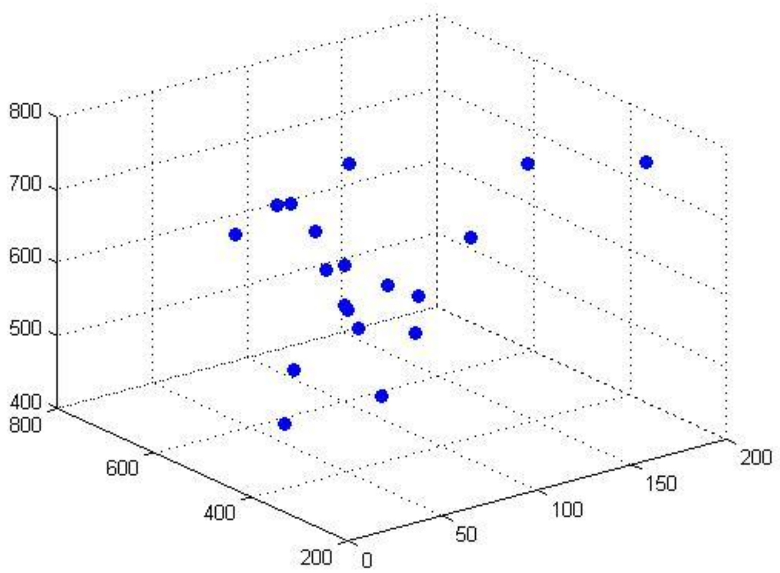
Szabálytalan elrendezésű pontok esetében jóval bonyolultabb interpolációt végezni. Ellenben a polinomiális regressziónak nem követelménye a pontok szabályos elhelyezkedése. Ez utóbbi esetben viszont ügyelni kell, hogy nagyon magas fokszámú polinomot nem szabad alkalmazni a túltanulás miatt. Az illesztő pontokban a hibaösszeg csökkenése ugyanis nem jelent feltétlenül megbízhatóbb modellt, sokszor a pontok között a felületben nagyfokú hullámzás, oszcilláció lép fel.

Nézzünk egy példát kétváltozós regresszióra, interpolációra! A magyar Duna vízgyűjtőjének területei Ausztriában és Bajorországban találhatók. Ha itt jelentős mennyiségű csapadék esik, akkor a Dunán árhullám vonul le. Feladatunk a budapesti tetőző vízállás előrejelzése a következő két adat alapján:

- x: az árhullámot kiváltó csapadék, amely 15 bajor illetve osztrák csapadékjelző állomás adatainak középértéke [mm];
- y: a Duna vízállása Budapestnél az árhullámot kiváltó esőzések kezdetekor [cm];
- z: a becsülendő érték Budapestnél, az árhullám tetőzéséhez tartozó vízszint [cm].

Az eddig mért adatokat az alábbi táblázat tartalmazza (az adatok az arhullam.txt file-ban elérhetőek), ebben egy sor egy adott időponthoz tartozó x, y, z értékeket jelenti:

58	405	590
52	450	660
133	350	780
179	285	770
98	330	710
72	400	640
72	550	670
43	480	520
62	450	660
67	610	690
64	380	500
33	460	460
57	425	610
62	560	710
54	420	620
48	620	660
68	390	620
74	350	590
95	570	740



2017. augusztus 3-án az osztrák és bajor vízgyűjtő területen átlagosan 100 mm csapadék esett. Ugyanekkor a Duna vízszintje Budapesten 400 cm-nél volt. Mekkorára várható az árhullám tetőzése Budapesten?

[2] Felhasználva: Paláncz Béla (2012): Numerikus módszerek példatár + előadás fóliák

Illesszünk egy másodfokú regressziós polinomot a pontokra, és ennek felhasználásával adjunk becslést a várható tetőzési magasságra! Illesszünk felületet a pontokra interpolációval is, lineáris és spline interpolációt alkalmazva és ezek alapján is becsüljük meg az árhullám várható tetőzését!

Először olvassuk be és ábrázoljuk az adatokat! 3D szórt pontokat a **plot3** vagy a **scatter3** paranccsal tudunk megjeleníteni.

```
> clc; clear all; close all;
> xyz=load('arhullam.txt')
> x=xyz(:,1); % árhullámot kiváltó csapadék [mm]
> y=xyz(:,2); % Duna vízállása Budapestnél az esőzés kezdetekor [cm]
> z=xyz(:,3); % a becsülendő érték, árhullám tetőzése Budapestnél [cm]
> % Ábrázolás
> figure(1);clf;
> scatter3(x,y,z,'filled')
```

Kétváltozós esetben regressziós síkot a következő módon írhatunk fel:

$$z = f(x,y) = p_1 + p_2 x + p_3 y$$

A másodfokú regressziós polinom általános felírása pedig a következő:

$$z = f(x,y) = p_1 + p_2 x + p_3 y + p_4 x^2 + p_5 xy + p_6 y^2$$

Harmadfokú regressziós polinom:

$$z = f(x,y) = p_1 + p_2 x + p_3 y + p_4 x^2 + p_5 xy + p_6 y^2 + p_7 x^3 + p_8 x^2 y + p_9 x y^2 + p_{10} y^3$$

Hasonlóképp negyed-, ötöd- stb. fokú polinomokat is felírhatnánk, de a túltanulás jelensége miatt 5. fokúnál magasabbat nem szoktak alkalmazni.

Most a feladatban másodfokú polinomiális felületet kell illesztenünk. n pont esetén n darab lineáris egyenletet írhatunk fel, és 6 meghatározandó paraméterünk van. Ez mátrixos alakban a következő lesz ($A \cdot p = b$):

$$A = \begin{pmatrix} 1 & x_1 & y_1 & {x_1}^2 & x_1 y_1 & {x_1}^2 \\ 1 & x_2 & y_2 & {x_2}^2 & x_2 y_2 & {x_2}^2 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & x_n & y_n & {x_n}^2 & x_n y_n & {x_n}^2 \end{pmatrix}; p = \begin{pmatrix} p_1 \\ p_2 \\ \vdots \\ p_6 \end{pmatrix}; \ b = \begin{pmatrix} z_1 \\ z_2 \\ \vdots \\ z_n \end{pmatrix}$$

Mivel a feladatban jóval több, mint 6 adatunk van, ezért egy túlhatározott lineáris egyenletrendszert kell megoldanunk, a legkisebb négyzetek módszerével minimalizálva a maradék ellentmondásokat. A feladat megoldása lényegében ugyanúgy megy, mint egyváltozós esetben.

```
> n = length(x) % 19
> A = [ones(n,1) x y x.^2 x.*y y.^2]
> p = A\z
```

A megoldást megkaphatjuk a matlab egy másik beépített függvényét, a **regress**-t használva is, ekkor az együtthatók mellett megkaphatjuk a 95 százalékos konfidencia intervallumukat és a maradék ellentmondásokat is

```
> [p int95 err]=regress(z,A)
```

Definiáljuk a regressziós felületet függvényként, és rajzoljuk be az illesztett felület az ábrába! Ábrázoljuk szintvonalakkal is a felületet! Becsüljük meg a várható árhullám tetőzését 100 mm csapadék és 400 cm-es vízszint esetén!

```
> f=@(x,y) p(1)+p(2)*x+p(3)*y+p(4)*x.^2+p(5)*x.*y+p(6)*y.^2
> hold on;
> fsurf(f,[min(x),max(x),min(y),max(y)])
```

Forgassuk el kicsit az ábrát a Rotate 3D paranccsal a grafikus ablakban, hogy jobban látszódjon az illeszkedés. Ábrázoljuk az illesztett *f* függvényt szintvonalas ábrán is! Függvény esetében a **contour** parancs nem használható, hanem vagy az **ezcontour** vagy az **fcontour**. A Matlab 2017-es verziójától az **ezcontour** használata nem javasolt, hanem helyette az **fcontour**, viszont ez utóbbi esetében egyelőre nem működik a szintvonalak feliratozása, úgyhogy egyelőre maradjunk az előbbinél. Ha a feliratokat be szeretnénk kapcsolni, akkor a szintvonalas ábrát el kell mentin egy változóba, és utána a **set** parancs segítségével állíthatjuk be a feliratozást ('Show','on'), illetve itt adhatjuk meg, hogy melyik szintvonalak jelenjenek meg ('LevelList',450:50:800). Figyeljünk, hogy a sima contour parancs esetében a ('ShowText','on') kapcsolót kellett beállítani, itt ez eltér ettől.

```
> figure(2); hold on;
> h = ezcontour(f, [min(x) max(x) min(y) max(y)])
> set(h,'Show','on','LevelList',450:50:800)
```

Az utolsó paranccsal beállítottuk, hogy a szintvonal feliratai is jelenjenek meg, és, hogy az alap 100 m helyett 50 m-ként rajzoljon szintvonalakat.

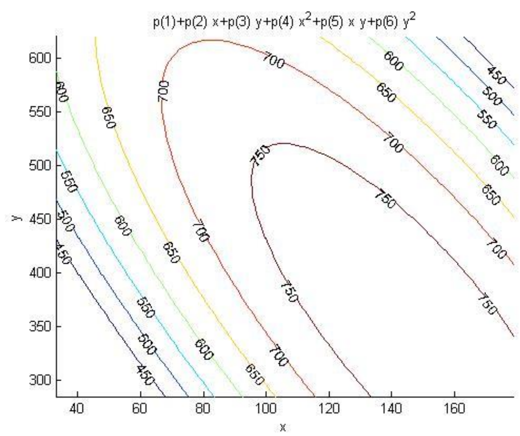
Megj. A legújabb Matlabnál javasolják az ezcontour helyett az fcontour használatát. Itt sajnos egyelőre nincs lehetőség feliratozni a szintvonalakat.

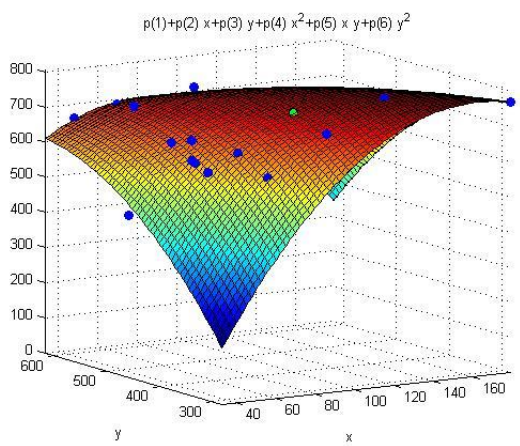
```
> fcontour(f, [min(x) max(x) min(y) max(y)],'LevelList',450:50:800)
```

Az árhullám várható tetőzése 100 mm csapadék és 400 cm-es vízszint esetén a regressziós polinommal végzett becslés alapján 738 cm lesz:

```
> % Előrejelzés
> zpol = f(100,400) % 737.8771
```

Az első ábránkba, ahol az adatokat megjelenítettük rajzoljuk be az illesztett felületet 3D-ben és az előre jelzett értéket is. Forgassuk el kicsit az ábrát a Rotate 3D paranccsal a grafikus ablakban, hogy jobban látszódjon az illeszkedés.



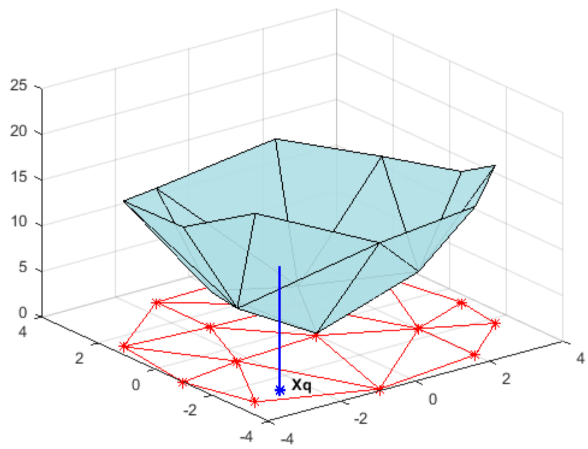


KÉTVÁLTOZÓS INTERPOLÁCIÓ SZABÁLYTALAN ELRENDEZÉSŰ PONTOK ESETÉN

Szabálytalan elrendezésű pontok esetében többféle interpolációs módszert szoktak alkalmazni. Lehet ez pl. a legközelebbi szomszéd módszere, amikor a pont a hozzá legközelebb eső pont értékét kapja, történhet az interpoláció krigeléssel, távolsággal fordított súlyozással, szabályos geometriai elemekre bontással vagy radiál–bázisfüggvény módszerét alkalmazva.

A Matlab beépített függvényei között a szabályos geometriai elemekre bontást találjuk meg. A **griddata** függvény Delaunay háromszög hálózatot vesz fel, és a háromszögek alapján végez interpolációt. Az interpoláció típusa lehet legközelebbi szomszéd interpoláció ('nearest'), háromszög alapú lineáris interpoláció, ekkor minden háromszögre egy síkot illesztünk ('linear'), vagy háromszög alapú köbös interpoláció ('cubic' - háromszögelésen alapuló 'bicubic spline' interpoláció). Illetve használható még a 'v4' módszer is, ami biharmonikus spline interpolációt végez. Ez utóbbi módszer nem háromszögelésen alapul.

A módszerek alapja a Delaunay háromszögelés, ahol úgy veszik fel a szórt pontok alapján a háromszögeket, hogy egy háromszög köré írt körbe ne kerüljön másik pont. Előnye ennek a háromszögelésnek, hogy mivel maximalizálja a háromszög legkisebb szögét, ezáltal kerüli a keskeny háromszögeket. Néhány alkalmazási területe: domborzatmodellezés (segítségével könnyen számíthatók az esés-, kitettség viszonyok, meghatározhatóak a szintvonalak), illetve a végeselem-módszerben is gyakran használják, mivel könnyen előállítható.



Határozzuk meg a háromszög alapú lineáris és köbös interpolációval is az árhullám várható tetőzését, ha a lehullott csapadék 100 mm és a vízállás Budapestnél 400 cm!

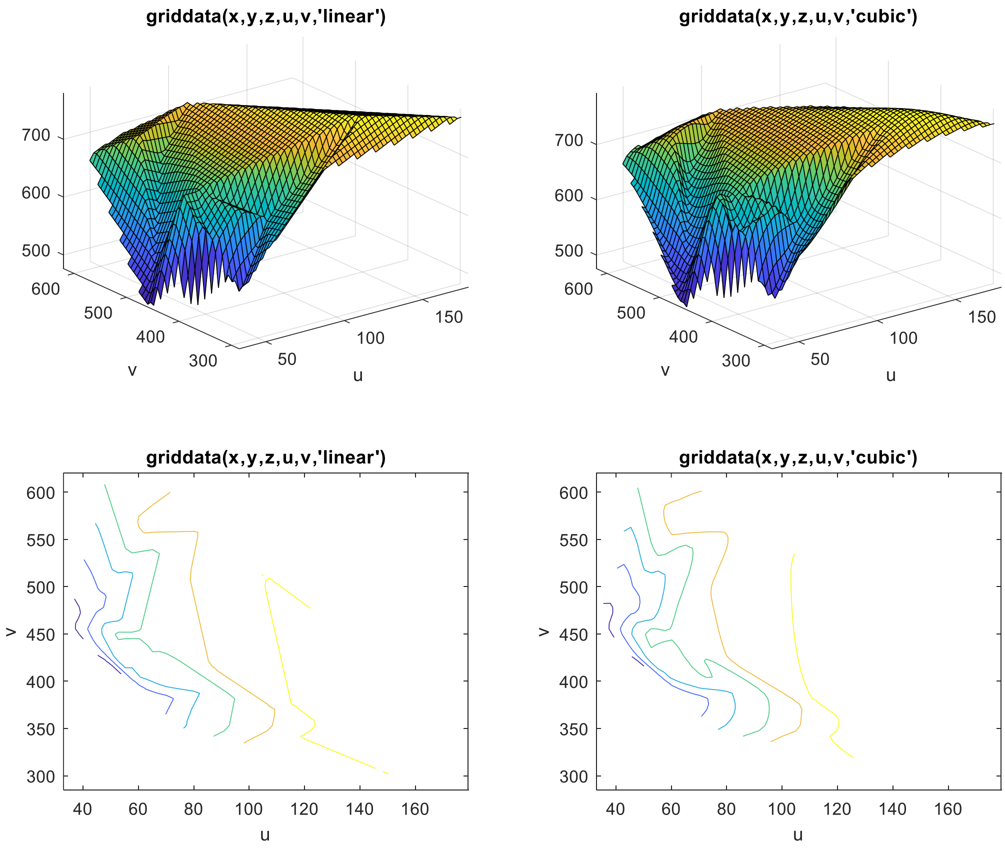
```
> zlin = griddata(x,y,z,100,400,'linear') % 724.7377
> zkob = griddata(x,y,z,100,400,'cubic') % 735.3939
```

A polinom illesztéssel végzett előrejelzés alapján a tetőzés 738 cm-en várható, a lineáris interpoláció alapján 725 cm-en, a köbös alapján pedig 735 cm-en. Függvényként is definiálhatjuk a fentieket, és akkor tetszőleges pontra könnyen meghívhatjuk, pl. a köbös interpolációra:

```
> % Interpolációs függvény definiálása
> flin = @(u,v) griddata(x,y,z,u,v,'linear')
> fkob = @(u,v) griddata(x,y,z,u,v,'cubic')
> flin(100,400) % 724.7377
> fkob(100,400) % 735.3939
```

Ábrázoljuk a két függvényt térben, illetve szintvonalakkal, egy ábra 4 részterületén. Sajnos a griddata-ból előállított függvényt az **fplot** nem jeleníti meg, csak az **ezplot**.

```
> figure(3);
> subplot(1,2,1); ezsurf(flin,[min(x) max(x) min(y) max(y)])
> subplot(1,2,2); ezsurf(fkob,[min(x) max(x) min(y) max(y)])
> subplot(2,2,3); ezcontour(flin,[min(x) max(x) min(y) max(y)])
> subplot(2,2,4); ezcontour(fkob,[min(x) max(x) min(y) max(y)])
```



ÚJ FÜGGVÉNYEK A GYAKORLATON

diff	-	vektor elemeinek különbsége, közelítő numerikus derivált, szimbolikus derivált számítása
cumsum	-	vektor elemeinek folyamatos összegzése
meshgrid	-	2-3 dimenziós rács előállítása vektorban tárolt x,y(,z) koordinátákból
plot3	-	Pontok 3D megjelenítése
mesh	-	Rácshálóban adott 3D pontok megjelenítése térbeli rácsként
surf	-	Rácshálóban adott 3D pontok megjelenítése színezett felületként (kitöltött térbeli rács)
contour	-	Rácshálóban adott 3D pontok alapján szintvonalak rajzolása
set	-	grafikus objektum (pl. h) megadott tulajdonságainak beállítása, pl. szintvonalak feliratozása (set(h,'ShowText','on') contour parancs esetén, vagy set(h,'Show',on) ezcontour esetén)

interp2	-	2D interpoláció rácshálóban adott pontokból tetszőleges pontra (módszer: lineáris – ’linear’, legközelebbi szomszéd - ’nearest’, spline inetrpoláció – ’spline’, 2D köbös spline (bicubic) – ’cubic’)
integral2	-	Kettős integrál számítása numerikusan, szabályos téglalap tartományon
scatter3	-	Szórt pontok 3D megjelenítése
regress	-	Többváltozós lineáris regresszió legkisebb négyzetek módszerével
fsurf, ezsurf	-	3D felületek kirajzolása megadott függvény alapján
fcontour, ezcontour	-	Szintvonalak kirajzolása függvény alapján
griddata	-	Interpoláció szórt pontok alapján tetszőleges pontra vagy rácsra (módszer: háromszög alapú lineáris interpoláció (TIN modell) – ’linear’, legközelebbi szomszéd - ’nearest’, háromszög alapú köbös interpoláció – ’cubic’, biharmonikus spline inetrpoláció – ’v4’)